Olist Ecommerce Analytics: User Guide

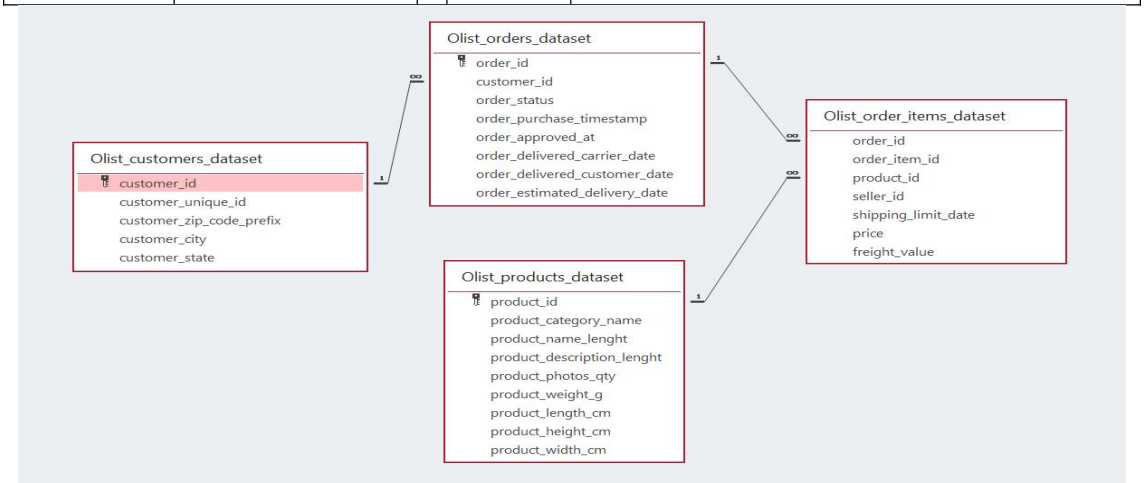
1. Introduction

This project is about an online shopping platform. We got the data of Olist, an e-commerce platform in Brazil from Kaggle, and used part of the data to create our own database based on our needs. Our project could be used to analyze sales performance and manage the platform data.

2. Database

2.1 Tables

Olist_customers_dataset table			Olist_orders_dataset table	
Primary Key	customer_id (ID generated per order)		Primary Key	order_id (unique per order)
customer_unique_id	unique ID for every customer		Foreign Key	customer_id (links to Olist_customers_dataset)
Other variables	customer location information		Other variables	order status, timestamps for various stages, and estimated delivery dates.
Olist_products_dataset table			Olist_order_items_dataset table	
Primary Key	product_id		Foreign Key	order_id (links to Olist_orders_dataset)
Other variables	product descriptions		Foreign Key	product_id (links to Olist_products_dataset)
			Other variables	order_item_id (item sequence in an order), seller IDs, shipping limits, prices, and freight values.



2.2 Queries

(1) Select TopX product by sales quantity
Merge [Olist_products_dataset], [Olist_order_items_dataset] and [Olist_orders_dataset].Filter for delivered orders, count sales per product, sort descending, and display the top X results.

```
strSQL = "SELECT Top " & TopI & " [Olist_order_items_dataset].[product_id], " & _
        "[Olist_products_dataset].[product_category_name], " & _
        "Count([Olist_order_items_dataset].[product_id]) AS ProductCount " & _
        "FROM [Olist_products_dataset] " & _
        "RIGHT JOIN ([Olist_order_items_dataset] " & _
        "LEFT JOIN [Olist_orders_dataset] " & _
        "ON [Olist_order_items_dataset].[order_id]=[Olist_orders_dataset].[order_id]) " & _
        "ON [Olist_order_items_dataset].[product_id]=[Olist_products_dataset].[product_id]" & _
        "Where [Olist_orders_dataset].[order_status]='delivered' " & _
        "GROUP BY [Olist_order_items_dataset].[product_id], [Olist_products_dataset].[product_category_name] " & _
        "ORDER BY Count([Olist_order_items_dataset].[product_id]) DESC;"
```

(2) Select TopX product by sales revenue
Similar to the above, but calculate total sales revenue per product (excluding freight value) and display top X results.

```
strSQL = "SELECT Top " & TopI & " [Olist_order_items_dataset].[product_id], " & _
        "[Olist_products_dataset].[product_category_name], " & _
        "Sum([Olist_order_items_dataset].[price]) AS ProductSales " & _
        "FROM [Olist_products_dataset] " & _
        "INNER JOIN ([Olist_order_items_dataset] " & _
        "INNER JOIN [Olist_orders_dataset] " & _
        "ON [Olist_order_items_dataset].[order_id]=[Olist_orders_dataset].[order_id]) " & _
        "ON [Olist_order_items_dataset].[product_id]=[Olist_products_dataset].[product_id]" & _
        "Where [Olist_orders_dataset].[order_status]='delivered' " & _
        "GROUP BY [Olist_order_items_dataset].[product_id], [Olist_products_dataset].[product_category_name] " & _
        "ORDER BY Sum([Olist_order_items_dataset].[price]) DESC;"
```

(3) Sales by year and month

Merge [Olist_order_items_dataset] and [Olist_orders_dataset]. Filter for delivered orders and exclude records with null delivery dates. Group by year and month based on delivery date.

```
strSQL = "SELECT YEAR([Olist_orders_dataset].[order_delivered_customer_date]) AS [Sales Year], " & _
         "MONTH([Olist_orders_dataset].[order_delivered_customer_date]) AS [Sales Month], " & _
         "Sum([Olist_order_items_dataset].[price]+[Olist_order_items_dataset].[freight_value]) As [Total Sales] " & _
        "FROM [Olist_order_items_dataset] " & _
        "INNER JOIN [Olist_orders_dataset] " & _
        "ON [Olist_order_items_dataset].[order_id]=[Olist_orders_dataset].[order_id] " & _
        "Where [Olist_orders_dataset].[order_status]='delivered' " & _
        "AND [Olist_orders_dataset].[order_delivered_customer_date] IS NOT NULL " & _
        "GROUP BY YEAR([Olist_orders_dataset].[order_delivered_customer_date]), " & _
        "MONTH([Olist_orders_dataset].[order_delivered_customer_date]);"
```

(4) Check customer order

Merge [Olist_orders_dataset] and [Olist_customers_dataset] and only display the related records of a specific customer.

```
strSQL = "SELECT [Olist_customers_dataset].[customer_unique_id], " & _
         "[Olist_orders_dataset].[order_id], " & _
         "[Olist_orders_dataset].[order_purchase_timestamp], " & _
         "[Olist_orders_dataset].[order_status] " & _
         "FROM [Olist_orders_dataset] " & _
         "INNER JOIN [Olist_customers_dataset] " & _
         "ON [Olist_orders_dataset].[customer_id]=[Olist_customers_dataset].[customer_id] " & _
         "WHERE [Olist_customers_dataset].[customer_unique_id] ='" & CustomerID & "';"
```

3. Front-end

On front-end we have one sheet as our home page. All the functions can be called by clicking on the tabs, the corresponding result will be shown on the same sheet.

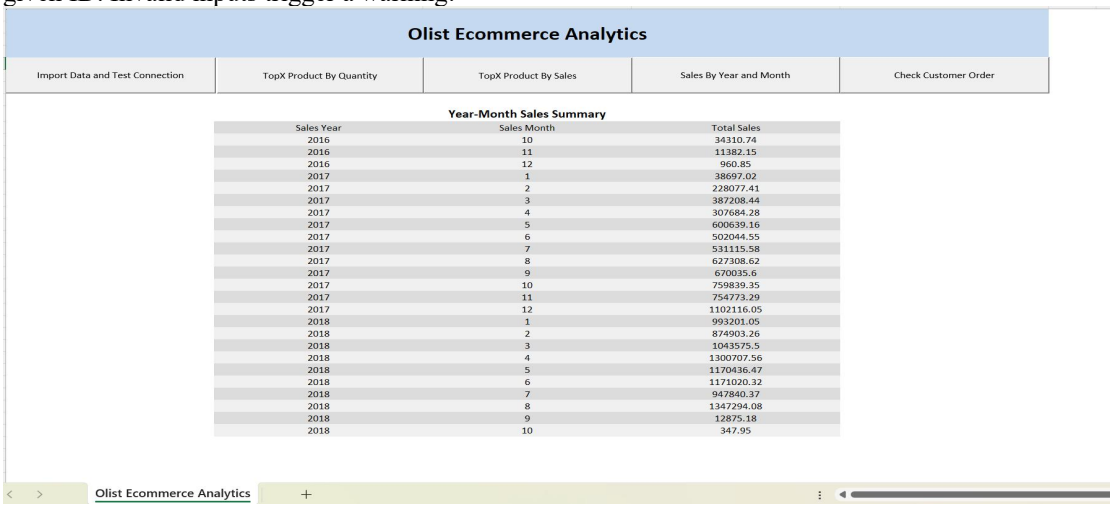
(1) Import data and test connection: Establishes a connection between Excel VBA and the Access database. Displays success/failure message.

(2) TopX product by sales quantity: Prompts user to input a number. Displays top X products by sales quantity. Invalid inputs trigger a warning.

(3) TopX product by sales revenue: Prompts user to input a number. Displays top X products by sales revenue. Invalid inputs trigger a warning.

(4) Sales by year and month: Displays a pivot table showing total sales by year and month.

(5) Check customer order: Prompts user to input a customer_unique_id. Displays orders related to the given ID. Invalid inputs trigger a warning.



4. VBA Middleware

4.1 Database Connection

We define some public variables in order to use them in several subroutines.Then we define a function to open database connection to our Access database Olist.

```
Option Explicit
Public rsResults As ADODB.Recordset
Public OlistDataBase As ADODB.Connection
Public ws As Worksheet
Public strSQL As String
Public Title As String
Function Open_Database_Connection() As ADODB.Connection
    Dim strProvider As String
    Dim strDataSource As String
    strProvider = "Provider=Microsoft.ACE.OLEDB.12.0;"
    strDataSource = "Data Source=" & ThisWorkbook.Path & "\Olist.accdb"
    Dim strConnection As String
    strConnection = strProvider & strDataSource
    Dim dbConn As ADODB.Connection
    Set dbConn = New ADODB.Connection
    dbConn.Open strConnection
    Set Open_Database_Connection = dbConn
End Function
```

4.2 Query Execution

This subroutine opens connection, stores it for reuse, and verifies its success.

```
Function Open_Database_Connection() As ADODB.Connection
    Dim strProvider As String
    Dim strDataSource As String
    strProvider = "Provider=Microsoft.ACE.OLEDB.12.0;"
    strDataSource = "Data Source=" & ThisWorkbook.Path & "\Olist.accdb"
    Dim strConnection As String
    strConnection = strProvider & strDataSource
    Dim dbConn As ADODB.Connection
    Set dbConn = New ADODB.Connection
    dbConn.Open strConnection
    Set Open_Database_Connection = dbConn
End Function
```

We have designed a subroutine that selects data from a recordset and displays it in Excel in a predefined format, making it easier for future use.

```
Sub DisplayResults(Title As String)
    Dim i As Integer
    Dim rowCount As Integer
    Dim colCount As Integer
    OlistDataBase.Open
    Set rsResults = New ADODB.Recordset
    Set rsResults = OlistDataBase.Execute(strSQL)
    Set ws = ActiveSheet
    ws.Range("A5:" & ws.Cells(ws.Rows.Count, "Z").Address).Clear
    For i = 1 To rsResults.Fields.Count
        ws.Cells(10, i + 1).Value = rsResults.Fields(i - 1).Name
    Next i
    ws.Range("B11").CopyFromRecordset rsResults
    rowCount = ws.Cells(ws.Rows.Count, "B").End(xlUp).Row - 9
    colCount = rsResults.Fields.Count
    If rowCount > 0 Then
        ws.Range("B10").Resize(rowCount, rsResults.Fields.Count).HorizontalAlignment = xlCenter
    End If
    Dim r As Integer, c As Integer
    For r = 10 To 9 + rowCount
        For c = 2 To colCount + 1
            If (r - 11) Mod 2 = 0 Then
                ws.Cells(r, c).Interior.Color = RGB(240, 240, 240) ' light grey
            Else
                ws.Cells(r, c).Interior.Color = RGB(220, 220, 220) ' dark grey
            End If
        Next c
    Next r
    With ws.Range(ws.Cells(9, 2), ws.Cells(9, 1 + colCount))
        .Clear
        .Merge
        .Value = Title
        .Font.Bold = True
        .Font.Size = 14
        .HorizontalAlignment = xlCenter
        .VerticalAlignment = xlCenter
    End With
    rsResults.Close
    Set rsResults = Nothing
    OlistDataBase.Close
End Sub
```

This subroutine prompts the user for a number(if input is invalid a warning will pop up), validates the input, executes the SQL to select the Top X products by sales quantity, and displays the records.

```
Sub TopProductByQuantity()
    Dim TopI As Integer
    TopI = Application.InputBox("Please enter the number of top sales data to display:", "Enter TopI", Type:=1)
    If TopI <> Int(TopI) Or TopI <= 0 Or TopI > 1000 Then '
        MsgBox "Please enter a valid positive integer (1 to 1000)!"
        Exit Sub
    End If
    Title = "Top " & TopI & " Product Sales by Quantity"
    strSQL = "SELECT Top " & TopI & " [Olist_order_items_dataset].[product_id], " & _
             "[Olist_products_dataset].[product_category_name], " & _
             "COUNT([Olist_order_items_dataset].[product_id]) AS product_sales_quantity " & _
             "FROM [Olist_products_dataset] " & _
             "INNER JOIN ([Olist_order_items_dataset] " & _
             "INNER JOIN [Olist_orders_dataset] " & _
             "ON [Olist_order_items_dataset].[order_id]=[Olist_orders_dataset].[order_id]) " & _
             "ON [Olist_order_items_dataset].[product_id]=[Olist_products_dataset].[product_id]" & _
             "Where [Olist_orders_dataset].[order_status]='delivered' " & _
             "GROUP BY [Olist_order_items_dataset].[product_id], [Olist_products_dataset].[product_category_name] " & _
             "ORDER BY COUNT([Olist_order_items_dataset].[product_id]) DESC;"
    DisplayResults (Title)
End Sub
```

This is pretty much like the above one, except that we display the TopX product by sales revenue.

```
Sub TopProductBySales()
    Dim TopI As Integer
    TopI = Application.InputBox("Please enter the number of top sales data to display:", "Enter TopI", Type:=1)
    If TopI <> Int(TopI) Or TopI <= 0 Or TopI > 1000 Then '
        MsgBox "Please enter a valid positive integer (1 to 1000)!"
        Exit Sub
    End If
    Title = "Top " & TopI & " Product Sales by Revenue"
    strSQL = "SELECT Top " & TopI & " [Olist_order_items_dataset].[product_id], " & _
             "[Olist_products_dataset].[product_category_name], " & _
             "Sum([Olist_order_items_dataset].[price]) AS product_sales_revenue " & _
             "FROM [Olist_products_dataset] " & _
             "INNER JOIN ([Olist_order_items_dataset] " & _
             "INNER JOIN [Olist_orders_dataset] " & _
             "ON [Olist_order_items_dataset].[order_id]=[Olist_orders_dataset].[order_id]) " & _
             "ON [Olist_order_items_dataset].[product_id]=[Olist_products_dataset].[product_id]" & _
             "Where [Olist_orders_dataset].[order_status]='delivered' " & _
             "GROUP BY [Olist_order_items_dataset].[product_id], [Olist_products_dataset].[product_category_name] " & _
             "ORDER BY Sum([Olist_order_items_dataset].[price]) DESC;"
    DisplayResults (Title)
End Sub
```

In this subroutine, we display total sales by year and month.

```
Sub AnnualandMonthlySales()
    strSQL = "SELECT YEAR([Olist_orders_dataset].[order_delivered_customer_date]) AS [Sales Year], " & _
             "MONTH([Olist_orders_dataset].[order_delivered_customer_date]) AS [Sales Month], " & _
             "Sum([Olist_order_items_dataset].[price]+[Olist_order_items_dataset].[freight_value]) As [Total Sales] " & _
            "FROM [Olist_order_items_dataset] " & _
            "INNER JOIN [Olist_orders_dataset] " & _
            "ON [Olist_order_items_dataset].[order_id]=[Olist_orders_dataset].[order_id] " & _
            "Where [Olist_orders_dataset].[order_status]='delivered' " & _
            "AND [Olist_orders_dataset].[order_delivered_customer_date] IS NOT NULL " & _
            "GROUP BY YEAR([Olist_orders_dataset].[order_delivered_customer_date]), " & _
            "MONTH([Olist_orders_dataset].[order_delivered_customer_date]);"
    Title = "Year-Month Sales Summary"
    DisplayResults (Title)
End Sub
```

This function can check whether a given ID exists in our current customer unique ID records.

```
Function CheckCustomerID(CustomerID As String) As Boolean
    Dim strSQL As String
    Dim IsValid As Boolean
    IsValid = False
    strSQL = "SELECT COUNT(*) FROM [Olist_customers_dataset] WHERE [customer_unique_id]='" & CustomerID & "'"
    OlistDataBase.Open
    Set rsResults = New ADODB.Recordset
    Set rsResults = OlistDataBase.Execute(strSQL)
    If rsResults.Fields(0).Value > 0 Then
        IsValid = True
    End If
    rsResults.Close
    Set rsResults = Nothing
    OlistDataBase.Close
    CheckCustomerID = IsValid
End Function
```

This subroutine prompts the user for a customer unique ID, validates it, executes the SQL, and displays the corresponding records.

```
Sub CustomerOrder()
    Dim CustomerID As String
    CustomerID = Application.InputBox("Please enter the customer unique ID you want to query:", "Enter CustomerUniqueID", Type:=2)
    If Not CheckCustomerID(CustomerID) Then
        MsgBox "The customer unique ID is invalid."
        Exit Sub
    End If
    strSQL = "SELECT [Olist_customers_dataset].[customer_unique_id], " & _
             "[Olist_orders_dataset].[order_id], " & _
             "[Olist_orders_dataset].[order_purchase_timestamp], " & _
             "[Olist_orders_dataset].[order_status] " & _
             "FROM [Olist_orders_dataset] " & _
             "INNER JOIN [Olist_customers_dataset] " & _
             "ON [Olist_orders_dataset].[customer_id]=[Olist_customers_dataset].[customer_id] " & _
             "WHERE [Olist_customers_dataset].[customer_unique_id] ='" & CustomerID & "';"
    Title = "Customer Order History"
    DisplayResults (Title)
End Sub
```

5. Conclusion

Although our project uses data from a real-world e-commerce dataset, the selected subset limits the scope of analysis. To scale this application for a real business, the following improvements could be implemented:

(1)Enhanced Data Integration: Include real-time data feeds, customer behavior analytics, and inventory tracking for deeper insights.

(2)Improved Performance: Replace Excel VBA with a modern tech stack, such as Python for back-end, and Tableau for front-end.

(3)Interactive Visualizations: Integrate tools like Power BI or Tableau to provide dynamic charts and drill-down analysis.

For more details, you can access the project repository on:

https://github.com/HUANGXINYU222/Olist-Ecommerce-Analytics

References:

Brazilian E-Commerce Public Dataset by Olist